Migrating from PDE to Bndtools in Practice
Meet The Speakers

Jochen Hiller
Developer Evangelist
Deutsche Telekom AG
Seasoned Java Developer
OSGi and IoT Evangelist
Eclipse Concierge Committer

Amit Kumar Mondal
Software Engineer
Deutsche Telekom AG
Passionate Java Developer
OSGi and IoT Enthusiast
Eclipse Kura Committer
About QIVICON
About QIVICON

Eclipse SmartHome based Home Gateway

Partner Applications
QIVICON
Eclipse SmartHome
OSGi Framework: OSGi R4.2
JavaSE Embedded 8 Compact 2/3
OS: Linux

REST API
ONEM2M
3RD PARTY EMULATION

RULE ENGINE
VOICE ENABLING
PERSISTENCE
GUI

ECLIPSE SMARTHOME CORE

THINGS / BINDING API
UPNP
MDNS
MQTT
SERIAL
PROTOCOL SUPPORT

DISCOVERY SERVICE
PROTOCOL BINDING
PRODUCT BINDING
CLOUD BINDING
SUB-SYSTEM BINDING
Agenda

01 PDE Pain Points

02 About QIVICON
## PDE Pain Points

<table>
<thead>
<tr>
<th>How</th>
<th>Issue</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Dependency Management</strong></td>
<td>• Complex Configuration</td>
</tr>
<tr>
<td>• Target Definition File</td>
<td>• Workaround to add bundles not present in p2 repos</td>
</tr>
<tr>
<td>• Tycho</td>
<td>• Additional Dependencies</td>
</tr>
<tr>
<td>• pom.xml</td>
<td></td>
</tr>
<tr>
<td><strong>Manifest-First Approach</strong></td>
<td>• Semantic Versioning: Manual and Error-Prone</td>
</tr>
<tr>
<td>• MANIFEST.MF</td>
<td>• No Automated Generation</td>
</tr>
<tr>
<td><strong>Run Configurations</strong></td>
<td>• Not reliable</td>
</tr>
<tr>
<td>• .launch</td>
<td>• Non-deterministic Behavior</td>
</tr>
<tr>
<td></td>
<td>• Difficult to maintain</td>
</tr>
</tbody>
</table>
Motivation

- Single Workspace leads to Smooth Induction to the Project
- Better Support of API based Development
- Explicit Dependency Management
- Higher Modularity Maturity
- Motivation
Agenda

01 About QIVICON

02 PDE Pain Points

03 Motivation

04 Comparison
## Comparison

### PDE vs Bndtools

<table>
<thead>
<tr>
<th>Feature</th>
<th>PDE</th>
<th>Bndtools</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bytecode Based Import Analysis</td>
<td>✗</td>
<td>✔</td>
</tr>
<tr>
<td>Instant JAR Creation</td>
<td>✗</td>
<td>✔</td>
</tr>
<tr>
<td>Default Annotation Support</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>Automated Semantic Versioning</td>
<td>✗</td>
<td>✔</td>
</tr>
<tr>
<td>One Project Multi Bundles</td>
<td>✗</td>
<td>✔</td>
</tr>
<tr>
<td>Default OBR Based Management</td>
<td>✗</td>
<td>✔</td>
</tr>
<tr>
<td>Automated Header Generation</td>
<td>✗</td>
<td>✔</td>
</tr>
<tr>
<td>Nested Workspace Structure</td>
<td>✔</td>
<td>✗</td>
</tr>
<tr>
<td>Custom Project Template Support</td>
<td>✗</td>
<td>✔</td>
</tr>
</tbody>
</table>
Setting Up - Workspace

A single directory with all its sub-directories and their files, similar to a git workspace.

Workspaces should be named according to the bundle symbolic names of its projects.

Valid if it contains a cnf file.

The advised model is to use a directory with a cnf/build.bnd file.
Bndtools uses repositories to manage dependencies at build and runtime.

In Bndtools 3.3, P2 Repository Plugin has been introduced.

```
-plugin.4.openHAB = \
    aQute.bnd.repository.p2.provider.P2Repository; \
    name = "openHAB"; \n    url = "${openhab.url}"

-plugin.5.jmDNS = \
    aQute.bnd.repository.p2.provider.P2Repository; \
    name = "jmDNS"; \n    url = "${jmDNS.url}"

-plugin.6.EclipseSmartHome = \
    aQute.bnd.repository.p2.provider.P2Repository; \
    name = "EclipseSmartHome"; \n    url = "${esh.url}"```
Maven Bnd Repository plugin provides a full interface to any type of Maven repositories (Nexus or Artifactory)

```-plugin.2.Runtime
aQute.bnd.repository.maven.provider.MavenBndRepository; 
snapshotUrl = "${snapshot.urls}";
releaseUrl = "${release.urls}";
noupdateOnRelease = false;
name = "Runtime";
multi = "jar, zip";
index = ${config.root}/runtime.maven```
Setting Up - Workspace

runtime.maven

```
org.eclipse.jetty.websocket:websocket-client:9.3.20.v20170531
org.eclipse.jetty.websocket:websocket-common:9.3.20.v20170531

# UI
com.qivicon.gateway:system.app.aurora:par:1.171.0.20180910092416
com.qivicon.gateway:system.app.config:par:1.171.0.201809100741
com.qivicon.gateway:system.app.pairwiz:par:1.159.0.201802050925

# ProSyst
com.prosyst:mbssh-bundles:zip:1.170.20
com.prosyst:firmware-bundles:zip:1.170.20
```

Contains a list of coordinates specifying an archive in a Maven revision

An archive is a ZIP, POM, JAR, or any other type of file
Setting Up - Workspace

Directory Structure

- `src` = `src/main/java`, `src/main/resources`
- `bin` = `${bin.directory}`
- `testsrc` = `src/test/java`
- `testbin` = `${testbin.directory}`
- `target-dir` = `target`

Project Specific Config

- `javac.source` = 1.8
- `javac.target` = `${javac.source}
- `javac.compliance` = `${javac.source}`
- `javac.debug` = on
- `javac.encoding` = UTF-8
- `-sources` = false
- `-resolve.effective` = resolve, active
- `-x-overwritestrategy` = gc
- `-dsannotations` = *
- `-dsannotations-options` = `version;minimum=1.2.0`
- `-metatypeannotations` = *
- `-metatypeannotations-options` = `version;minimum=1.2.0`
- `-nouses` = true
- `-nodefaultversion` = true

https://bnd.bndtools.org
The command will first parse the manifest to establish the BSN

Creates a Bnd project in the provided workspace with the symbolic name
Post Conversion

Generated Descriptor

# Remove next line and META-INF directory when you are going to modify this bundle
# and fix any issues. This line mimics the PDE manifest first behavior
# and voids many functions of bndtools
-manifest: META-INF/MANIFEST.MF

Private-Package
    com.qivicon.runtime.upnp, \
    com.qivicon.runtime.upnp.util, \
    com.qivicon.runtime.upnp.av

Bundle-Name
QIVICON-Application
QIVICON-ApplicationCategory
-workingset
testsrc
-buildpath: \
    com.google.gson,\n    org.jupnp,\n    org.jupnp.support,\n    com.qivicon.service.hl,\n    com.qivicon.service.gson,\n    com.qivicon.service,\n    com.qivicon.smarthome.config.system,\n    com.qivicon.runtime
-testpath: \
    com.google.guava,\n    org.mockito.mockito-core,\n    org.apache.commons.lang,\n    net.bytebuddy.byte-buddy,\n    org.objenesis

Persist PDE Manifest

Automated Analysis

Choice of existing MANIFEST Usage should be configurable in Bnd
Timeline

Ideation Phase for Migration
- 2018 April

Research/Analysis
- 2018 May

Migration Trials
- 2018 June
Coping with the Current System: Finding Solutions

2018 July

Finalizing Solutions: Milestone Set for Migration

2018 August

Final Migration

2018 Early September
Challenges

**Multi-Project Sync**
- Integrating PDE with Bnd Projects together makes Eclipse confused

**Integration Testing**
- Bnd ignores Fragments but most PDE users write tests in Fragments

**Complex Infrastructure**
- Bnd cannot easily handle complex maven based infrastructure, needed tweaks, unable to process client certs

**Windows Environment**
- The development is bit slower on NTFS based systems.
  - Gradle also doesn’t work as efficient as it works in Unix based platforms.
Insider Tips

01. Organize Trainings and Hackathons

02. Adapt Tests during Migration

03. Disable Antivirus Scanning in: Eclipse, Workspace, .m2, .p2, .gradle, .groovy, .bnd, .eclipse

04. Learn BND Commands
Credits

01. Colleagues for Valuable Feedbacks
- Peter Kriens

02. Jan Hendriks

03. Alexander Hoffman

04. Benedikt Niehues

05. Bndtools Community
Evaluate the Sessions

Sign in and vote at eclipsecon.org

-1 0 +1
Thank You :)

[Image]